

General Environment for Human Interaction with a Robot Hand-Arm System and Associate Elements

Jose Fortín and Raúl Suárez

Abstract—Software development in robotics is a complex task due to the existing heterogeneity in terms of hardware, communications, and programming languages that are used in current robotic systems. In this work a general environment for the interaction between the human operator and different elements in a robotized cell is presented, such that all the involved elements can be easily managed from a unique interface. The aim of the developments is to provide a common frame that can be ported to different operating systems and can be easily extended or adapted to new devices.

I. INTRODUCTION

Human-robot interaction (HRI) is a multidisciplinary field that includes areas such as artificial intelligence, human-computer interaction, robotics or speech recognition, among several others. Human-computer interaction plays an important role in the development of HRI systems, researchers must test extensively the systems developed in order to put them in the same environment as humans, therefore providing a set of tools that aim to facilitate experimentation and interaction with the hardware components involved is extremely useful. Experimentation is an important part in research, simulation and visualization tools allow researchers to test their work at different stages of development, therefore this type of tools is an important addition to any system, as an effort to close the gap between the advances made in research robotics and the actual state of industrial applications.

In this work a general environment for the interaction between the human operator and different elements in a robotized cell is presented. The aim of the developments is the easy connection and interaction with different components through a common interface, with an intuitive graphical interaction. The final goal is to facilitate the interaction with the hardware components used

This work was partially supported by the CICYT projects DPI2007-63665 and DPI2008-02448.

The authors are with the Institute of Industrial and Control Engineering - Technical University of Catalonia, Barcelona, Spain. jose.antonio.fortin@estudiant.upc.edu, raul.suarez@upc.edu

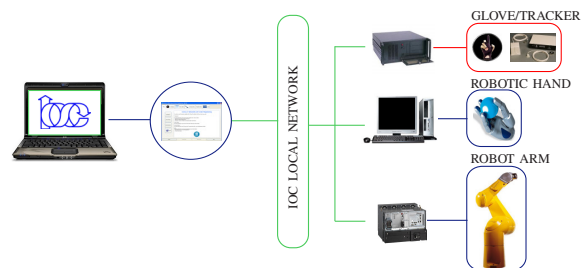


Fig. 1: Initial scenario. A set of hardware components are available but no common interface for interaction.

in the Robotics Laboratory of the IOC. After this introduction, the rest of this paper is organized as follows: Section II presents the initial scenario, the requirements established for development and a brief description of the different hardware components involved. Section III describes the used tools and the developments done in this work. Finally, Section IV presents a set of tests performed to validate the functionality of the software developments.

II. PROBLEM STATEMENT

The scenario is composed by the hardware components shown in Figure 1:

- A sensorized glove.
- A position tracking system.
- A robot arm.
- A robotic hand.

Currently, there are no general applications that allow the combined use of the systems mentioned above. In order to interact with each device, the user must write his/her own software applications to perform tasks such as collect the data from the devices or command the position of the robot arm and hand, this not only delays the process of experimentation but, in some cases, obliges the developer to learn a specific programming language to make this process feasible.

A. Requirements

The requirements established for this work are:

- The software architecture should be open and extensible.

- The developed application should provide a basic set of tools including: data visualization, combined use of different devices, a network communications protocol to operate the robotic hand and manipulator arm from a remote computer and the integration of the data glove with an existing simulator of the robotic hand.

B. Involved Hardware

The following four different hardware systems must be operated from the application:

- 1) *Sensorized Glove.* This device is a data glove part of the CyberGlove® system used to capture the human hand movements [1]. The particular glove used in this work has 22 sensors placed at critical points to measure the posture of the hand as shown in Figure 2(a). The data glove is connected to an interface unit, shown in Figure 2(b), that is in charge of translating the voltage output received from each sensor into digital values within the range [0, 255]. The interface unit uses a serial port to transmit the data collected from the data glove. The data rate can be up to 90 records/sec. The sensors resolution is 0.5 degrees and its repeatability is of 3 degrees which is also the typical standard deviation between different glove wearings.
- 2) *Tracking System.* The tracking system used in this work is the Flock of Birds [2], a 6 degrees of freedom measuring device that can be configured to simultaneously track the position and orientation of multiple sensors. Figure 3 presents the different elements that form the tracking system. The sensor measures the electromagnetic field produced by the transmitter; the strength of the received signals is compared to the strength of the sent pulses to determine the position, and the received signals are compared to each other to determine the orientation. The sensor resolution is 0.5 mm (positional) and 0.1 degrees (angular) both at 30.5 cm from the transmitter.
- 3) *Industrial Robot.* The industrial robot is an articulated robot arm with 6 degrees of freedom, model TX-90 manufactured by Stäubli [3]. The maximum payload for the manipulator arm is 20Kg. The robot arm system is composed of the four elements shown in Figure 4. In order to interact with the robot system, an application written in specific language (VAL3) [4] must be loaded into the controller.
- 4) *Robotic Hand.* The robotic hand used in this work is the Schunk Anthropomorphic Hand (SAH), the hand system is composed of the elements shown in Figure 5. The robotic hand has four

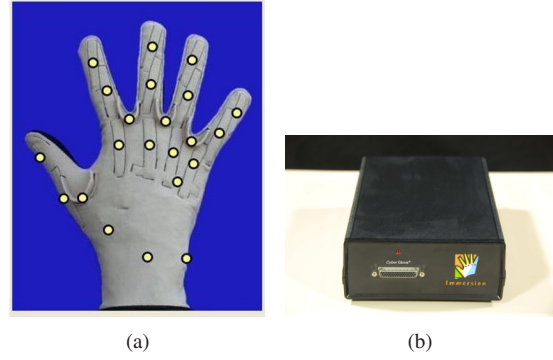


Fig. 2: Cyberglove system. (a) Data glove sensor disposition, (b) Interface unit.

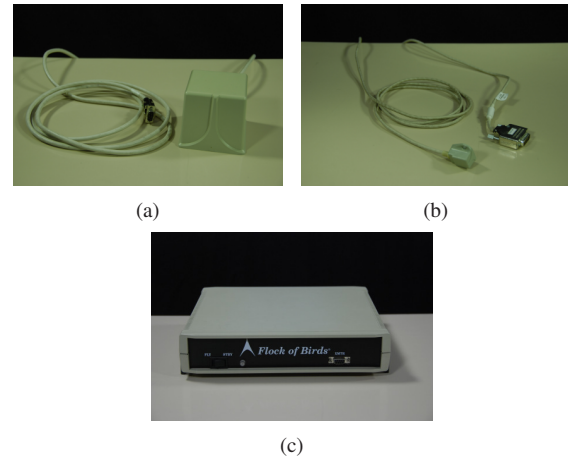


Fig. 3: Flock of Birds tracking system. (a) Transmitter, (b) Sensor, (c) Processing unit.

fingers, each finger has 4 joints and 3 independent degrees of freedom. The thumb finger base has an extra degree of freedom for power grasping and fine manipulation [5]. Instructions are sent to the robotic hand via the PCI card shown in Figure 5(b) which is installed in a personal computer (PC) in the Robotics Lab. The communication between both devices is done using a point to point serial communication (PPSeCo) system.

C. Proprietary Software Packages

For each system the manufacturers provide a set of software tools to facilitate interaction with their hardware components. However these are limited to a set of basic features.

- *CyberGlove Development library.* The development library is a set of routines written in C++ programming language, this library allows to collect the data provided by the data glove and third party hardware components such as the electromagnetic field position trackers from the Ascension Technologies [6] and Polhemus [7].

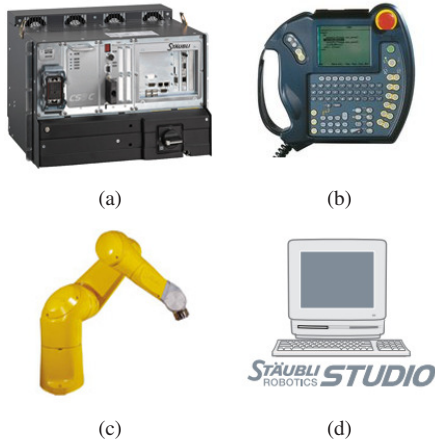


Fig. 4: Stäubli robot system. (a) CS8C controller, (b) Teach pendant, allows control of the robot locally, (c) Manipulator arm, (d) Stäubli Studio software workbench.

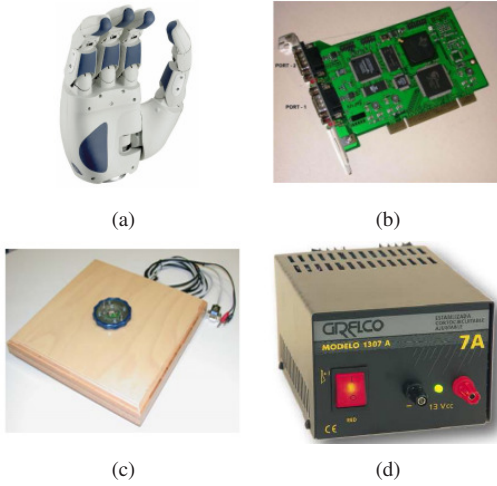


Fig. 5: Robotic hand system. (a) Robotic hand, (b) PCI card, (c) Mounting base, (d) Power supply.

- Flock of Birds. To collect the readings provided by the positional tracker, the development library for the data glove is used.
- SAH Application Programming Interface (API). This library written in C++ allows to perform actions such as controlling the angular position of the 13 joints of the robotic hand, retrieve the position of the joints and also the torque readings of the 12 sensors mounted in the hand.
- Stäubli Studio (manipulator arm). This software workbench is used to develop programs in the VAL3 language. Other features include a transfer tool to load the created applications into the CS8C controller in order to be executed by the robot arm system.

III. DEVELOPED SOLUTION

The developed solution structure is presented in Figure 6. This structure presents three main independent modules devote to communications, simulations and interaction with the hand, including each of them a graphical interface. Each of these modules is describe below in this section. The software developments done in this work use a set of open source packages written in C++. The most remarkable characteristics of this programming language are object-oriented programming, portability and speed of execution. The selected packages for development are Qt, Coin3D and SoQt.

- Qt is a cross-platform, graphical application development toolkit that enables developers to compile and run applications on Windows, Mac OS X, Linux [8].
- Coin3D is an OpenGL based retained mode 3D graphics rendering library [9].
- SoQt allows a Coin scene graph to be rendered inside a Qt application [10].

The software packages selected are free of charge. The programming language is the same for the selected packages and the software provided by the manufacturers facilitating the integration process. Other software packages used in this work are those mentioned in Subsection II-C: The CyberGlove development library used for the data glove and tracker, Stäubli Studio to create the VAL3 application to communicate with the TX-90 robot, and the SAH API to communicate with the robotic hand controller.

A. Integration with the hardware components

In order to use the hardware components from within a common frame, the developed and proprietary software packages must be combined. For the data glove and tracker this process consists in using three different classes from the CyberGlove development library, the *vhtIOConn* class, which handles the communication with both input devices, and the *vhtCyberGlove* and *vhtTracker* classes, which provide access the data glove and tracker features, respectively. For the robotic hand the *CSAHandCtrlApi* class (part of the SAH API) is used in order to access the available features of the robotic hand. These libraries are written in C++ and allow local access to the features of the devices, as shown in Figure 7. The robot arm can not be accessed as the rest of the devices, however, the operation of this device is possible through the use of networking sockets.

The type of sockets used for communication with the TX-90 robot are TCP sockets using a

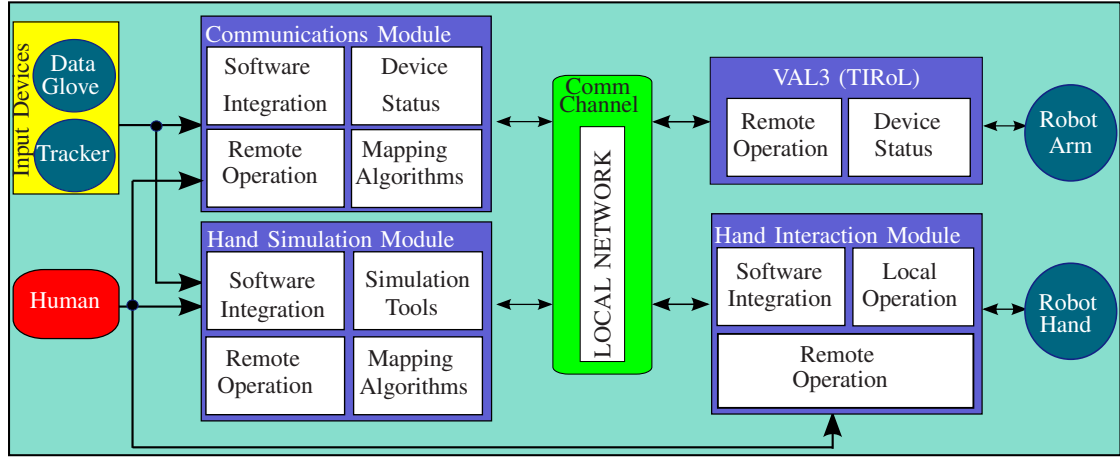


Fig. 6: Experimental Platform.

client-server model. The same approach has been used in [11] and [12] which address the problem of software integration in robotic systems. In order to use sockets to communicate two applications using a client-server model a set of steps must be performed, as shown in Figure 8. For the networking communication process the QtNetwork module has been used, the decision of using this module instead of following a traditional C++ implementation is that this module makes communications independent of the OS for the developer. The socket creation and initialization process is the same and independent of the OS used. The implementation of a client-server model has been used not only to communicate with the robot arm but to provide the user with means to interact with the other devices from a remote computer.

B. General layout

The solution presented in this work consists in the development of three independent modules capable of communicate and exchange information between them (Figure 6). A modular development (instead of a single application that handles all the devices) presents the following advantages:

- By developing smaller applications, new features can be added to each module without affecting the features already developed.
- Smaller applications are more sustainable, allowing the developer to find and correct errors faster.
- All modules can be used as stand-alone applications, making it possible for different users to work with the devices separately or combined.

The three modules are addressed in this paper as the communications module, the hand interaction

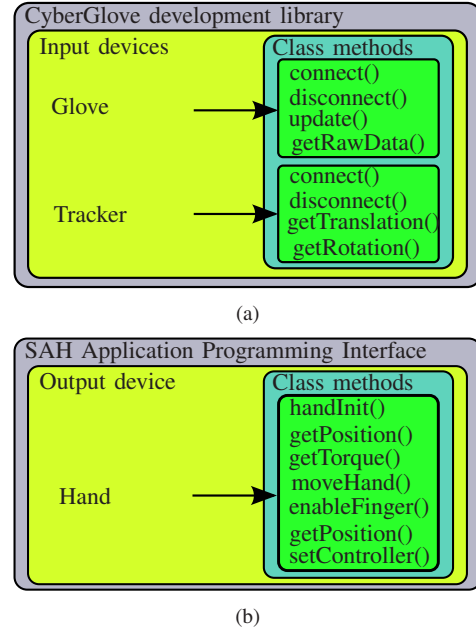


Fig. 7: C++ object definition. Integration of the devices is done using the proprietary software libraries as part of the application, in order to access their features.

module and the hand simulation module. They are described in the following subsections.

C. Communications Module

The communications module shown in Figure 9 provides communication with the input and output devices, i.e users can collect data from the input devices (data glove and tracker) or send movement commands to the output devices (robot arm and robotic hand).

- 1) *Input Devices*. For the data glove and tracker the same features are available, the user can

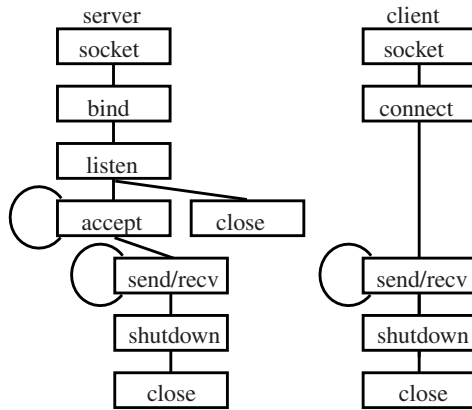


Fig. 8: TCP-Based sockets. Actions performed on each side in order to establish communication between both applications.

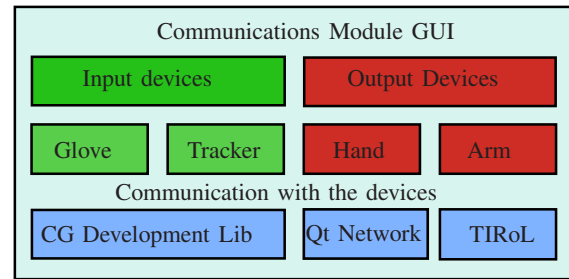
collect the data using two different modes of operation:

- **Time Intervals.** This mode allows the user to establish the period of time in which the application will collect the data from the device.
- **Sampling Mode.** This mode consists on saving specific hand configurations performed by the user wearing the glove or to collect the position and orientation of the tracking sensor.

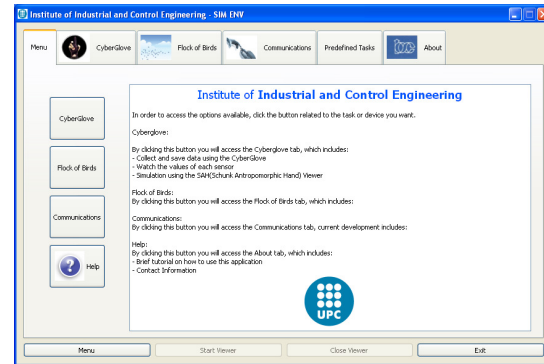
The user can retrieve the data from the different input devices separately or simultaneously. The data collected from each device can be stored to be used later.

- 2) **Output Devices.** For the robot arm and robotic hand, the features available consist in teleoperation and retrieving the information collected from each device, positional values for the robot arm and positional and torque values for the robotic hand. The communication with each output device is done as follows:

- **Robot arm:** In order to communicate with the TX-90 robot, a template written in VAL3 developed at the IOC Robotics Lab is used. Using this template inside a VAL3 application two different actions can be performed: send movement commands or retrieve the robot position. Each message is a text string containing the values of the six robot joints ($q_0, q_1, q_2, q_3, q_4, q_5$) or the Cartesian coordinates (x, y, z, R_x, R_y, R_z) of the robot tool base. Each value is separated by a space and the string terminator used is the '@' character to establish the end of the message.
- **Robotic Hand:** In order to communicate with the SAH, the network module implemented



(a)



(b)

Fig. 9: Communications Module. (a) Module structure, input devices are access through the CyberGlove development library and output devices using a client-server based model, (b) Developed graphical user interface.

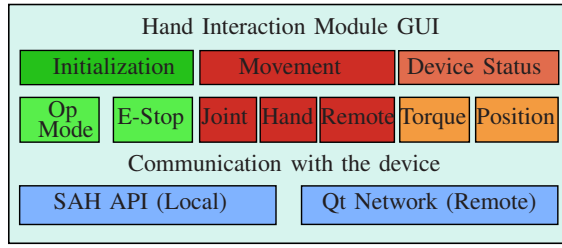
within the Qt toolkit is used. The messages sent to the robotic hand are constructed using a Data Stream structure. The sent message is composed of 14 values, the values are the 13 joint angle values and 1 value for the movement speed. However, independent speed values can be establish for 12 of the 13 joints, this is useful to produce a more human-like movement of the robotic hand.

D. Hand Interaction Module

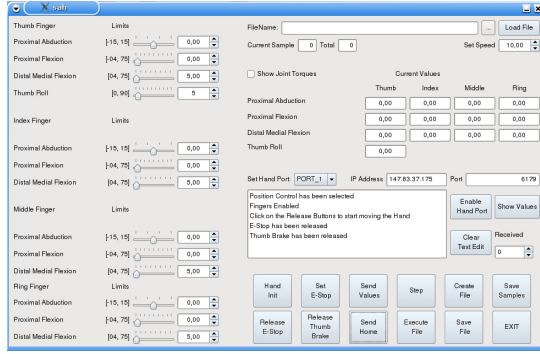
The hand interaction module shown in Figure 10 allows the user to perform the following tasks:

- Device initialization.
- Move each finger joint independently or perform hand configurations by moving all joints simultaneously.
- Remote connection and operation of the hand through the communications module.
- Retrieve information such as torque readings and the joint positions.
- Stop all movements in case of collision.

The hand interaction module is executed under Linux OS, so far no related work has been found that uses this robotic hand under Linux, and due to the possibility of developing cross-platform applications



(a)



(b)

Fig. 10: Hand Interaction Module. (a) Module structure, the hand features are accessed locally through the API and a client server model is used for remote operation, (b) Developed graphical user interface.

using Qt, this application could also be executed under Windows OS, by making some minor modifications to the source code.

1) *Device Initialization.* The initialization process consists of the following steps:

- Establish communication between the robotic hand and its controller.
- Set the mode of operation on which the robotic hand will work.
- Enable the hand fingers controllers.
- Revoke the brake signal sent to the fingers controllers.

For the final user, this procedure simply consists in clicking a button, all the steps mentioned above are handled internally.

- 2) *Hand Movement.* The hand movement can be accomplished in different ways, a user can have control over a single joint or have a predefined set of configurations, these values can be entered one by one or read as an input file and then converted to actual configurations performed by the hand.
- 3) *Collision Detection.* In order to detect possible collisions of the hand with the environment, the readings provided by the torque sensors are used, the sensors are placed after the gear-box of each joint, providing a precise measurement of the

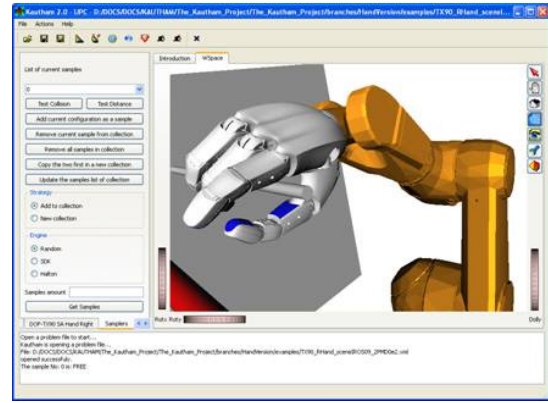


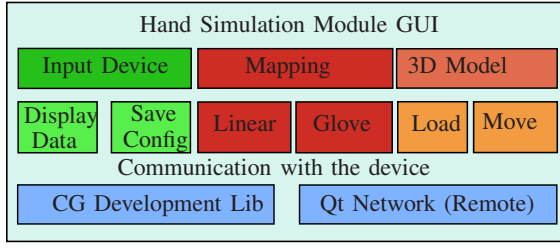
Fig. 11: Example of a simulation environment.

real joint torque. Based on the torque readings obtained from the device, a routine checks for possible collisions between fingers, if the value of a sensor surpasses the limits established, all pending movements are canceled and a message is displayed to inform the user. The information provided by the torque sensors can be used to detect whether the robotic hand has encountered an obstacle or it is properly manipulating an object.

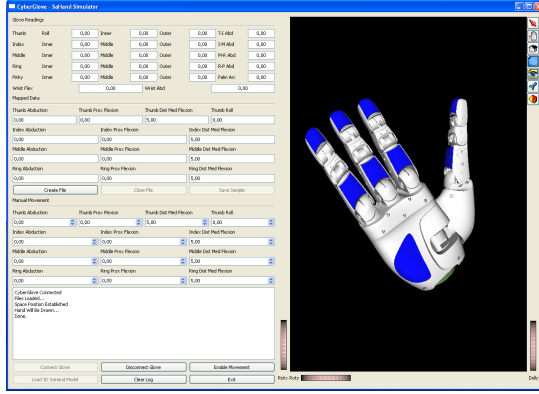
E. Hand Simulation Module

Simulation environments are used in a variety of fields, one of the advantages of using a simulator is that the users are not physically dependent of the device, making it possible to save costs and time. On the other hand new algorithms can be tested without the concern of damaging the real devices. Figure 11 shows an example of a simulation environment that is part of a path planning and collision detection toolkit developed at the IOC [13]. The hand simulation module shown in Figure 12 provides two main characteristics: a virtual representation of the robotic hand and a mapping implementation to move the virtual model using the data glove as input device.

- 1) *Virtual Hand Representation.* The virtual model of the robotic hand was created using the Coin3D library, which uses scene-graph data structures to provide visualization. A scene-graph consists of a set of nodes interconnected using a tree or hierarchical structure. Nodes can be used to represent groups, geometries or properties. All levels inside the scene-graph have at least one associated input file, that contains the information (geometry, color, texture, etc.) to create the 3D model of the different parts needed to build the robotic hand. The two other parameters used are the position and rotation, the first one is used to set where the object will be placed inside the



(a)



(b)

Fig. 12: Hand Simulation Module. (a) Module structure, (b) Developed graphical user interface.

scene-graph, and the rotation parameter allows to associate a rotation axis to this node.

- 2) *Joint-to-Joint Mapping*. Movement of the virtual model of the robotic hand is accomplished with the integration of the data glove. The input values received from the data glove are transformed into values inside the range of movement of the robotic hand. This transformation process is done using a joint-to-joint mapping. For each joint the information provided by a single sensor is used making it a one-to-one transformation. This method has been chosen due to its simplicity, however, the application allows the user to develop its own algorithms and integrate them into the simulation module. The mapped values are computed using the following parameters:

- Sensor gain, $G(i)$. Scale factor used to establish the relationship between the deformation of the sensors and the angle value for the joint i on the robotic hand,

$$G(i) = \frac{q_{UL}(i) - q_{LL}(i)}{\text{sensor}_{UL}(i) - \text{sensor}_{LL}(i)}$$

where $q_{UL}(i)$ and $q_{LL}(i)$ represent the upper and lower limits from each joint of the robotic hand listed in Table I, $\text{sensor}_{UL}(i)$ and $\text{sensor}_{LL}(i)$ represent the upper and lower limits in the range of movement accepted from

Joint	Thumb	Index	Middle	Ring
P_{min}	-4	-4	-4	-4
P_{max}	75	75	75	75
$M-D_{min}$	0	0	0	0
$M-D_{max}$	75	75	75	75
A_{min}	-15	-15	-15	-15
A_{max}	15	15	15	15
TR_{min}	0	—	—	—
TR_{max}	90	—	—	—

TABLE I: Range of movement for the robotic hand joints. P= Proximal, M-D= Middle-Distal, A= Abduction and TR= Thumb Roll.

Joint	Thumb	Index	Middle	Ring
$Inner_{min}$	81.09	103.90	76.57	100.0
$Inner_{max}$	186.80	199.90	142.90	193.0
$Middle_{min}$	25.36	48.39	69.48	63.18
$Middle_{max}$	178.70	198.20	157.90	162.40
Abd_{min}	139.20	93.96	—	75.24
Abd_{max}	189.95	177.80	—	152.70

TABLE II: Upper and lower limits accepted from the data glove sensors. The abduction of the middle finger is measured using the abduction sensors of the index and ring fingers.

each sensor in the glove as listed in Table II.

- Position offset, $Offset(i)$. To establish the offset parameter for each sensor, the “zero” or “home” position of the robotic hand is used as reference; the same pose is performed by the user wearing the data glove. The readings obtained from each sensor are then collected and used as the offset between the sensor lower accepted input values from the data glove according to the “zero” position established.

$$Offset(i) = \text{sensor}(i) - \text{joint}(i)$$

where $\text{sensor}(i)$ are the data glove readings and $\text{joint}(i)$ are the corresponding joint values of the hand.

After gain and offset parameters have been determined, the following equation is used to compute the mapped values,

$$q_{mapped}(i) = G(i) * (\text{sensor}_{raw}(i) - Offset(i))$$

where $q_{mapped}(i)$ represents the computed value for joint i of the robotic hand and $\text{sensor}_{raw}(i)$ parameter is the corresponding sensor value as read directly from the serial port.

IV. EXPERIMENTAL VALIDATION

In order to validate the features of each module, a set of tests have been performed. The tests include the mapping between the glove values and the robotic shown in Figure 13, teleoperation of the robotic hand using the data glove as input device shown in Figure 14 and Figure 15 shows a pick

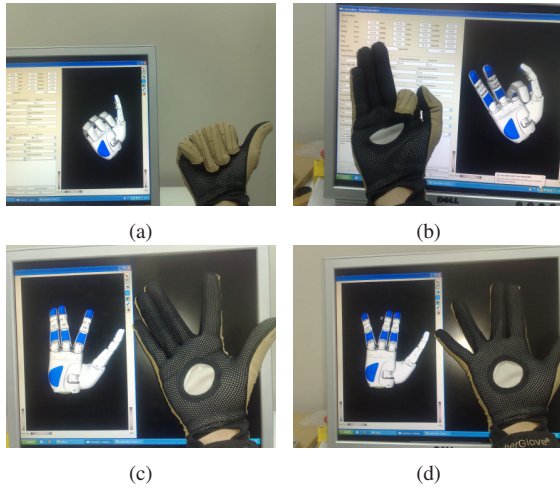


Fig. 13: Mapping results. (a) and (b) correspond to the fingers flexion movement, (c) and (d) correspond to the fingers abduction movement.

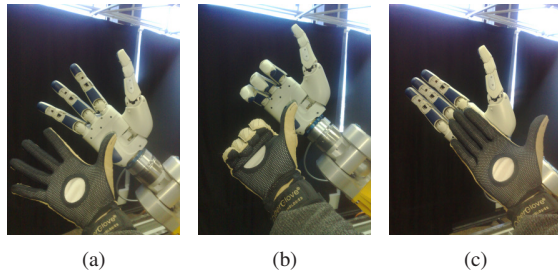


Fig. 14: Poses performed wearing the data glove and then sent to the robotic hand.

and place task that uses the robot arm and hand simultaneously.

V. CONCLUSIONS AND FUTURE WORK

The current implementation is being used actively in the Robotics Laboratory of the IOC to collect data from the input devices that is later used in grasping research. The simulation module is of great aid in order to test mapping algorithms for dexterous robotics hands. The combined use of the communication module and the hand interaction module provide a simple interface to command the movement of the robot arm together with the robotic hand. The modular design of this work facilitates the integration of different hardware components, since each module can be expanded separately without affecting the features already developed. The developed applications allow a simple use of the different hardware components by providing a multi-platform experimentation environment.

Short term future work consists in the expansion of the communications module by integrating

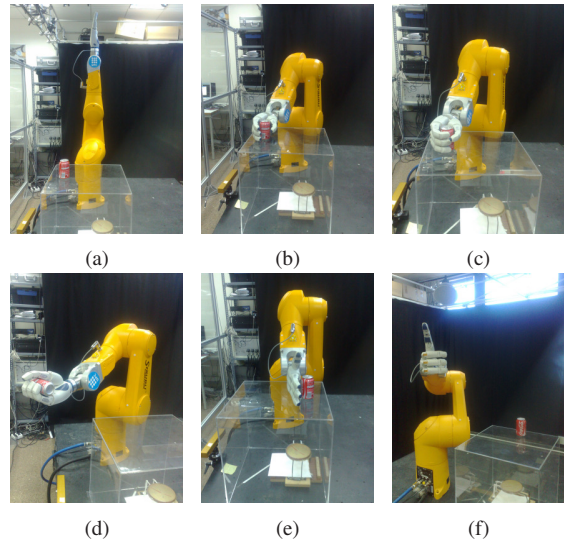


Fig. 15: Pick and place task performed by sending movement commands to both devices.

an exoskeleton to provide the user with a force feedback functionality. This device is intended to respond to the torque readings provided by the robotic hand through the Hand Interaction Module and by the Hand Simulation Module in order to determine whether a virtual object is being properly manipulated.

ACKNOWLEDGEMENTS

The authors would like to thank Leopold Palomo, Luca Colasanto, Carlos Rosales, Jan Rosell and Alexander Pérez for their collaboration in the development of this work.

REFERENCES

- [1] C. S. LLC., *CyberGlove Data Glove User Guide*, 2007.
- [2] A. T. Corporation, *Flock of Birds Installation and Operation Guide*, Jan. 2002.
- [3] S. Robotics, *Arm - TX series 90 family*, 2005.
- [4] S. Robotics, *VAL3 Reference Manual*, Jan. 2008.
- [5] D. German Aerospace Center, *Schunk Anthropomorphic Hand User Manual*, May 2007. Rev. 1.2.
- [6] "Ascension technology corporation." <http://www.ascension-tech.com/>.
- [7] "Polhemus." <http://www.polhemus.com>.
- [8] "Qt documentation." <http://qt.nokia.com/doc/>.
- [9] "Coin3d documentation." <http://doc.coin3d.org/Coin/>.
- [10] "Soqt documentation." <http://doc.coin3d.org/SoQt/>.
- [11] M. H. S. Fleury and R. Chatila, "Genom: A tool for the specification and the implementation of operating modules in a distributed robot architecture," *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 842–848, Sep. 1997.
- [12] A. M. Alex Brooks, Tobias Kaupp and S. Williams, "Towards component-based robotics.," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.
- [13] A. Pérez and J. Rosell, "A roadmap to robot motion planning software development," *Journal of Intelligent and Robotic Systems*, vol. 53, pp. 223–245, Nov. 2008.